AuthorWords

October 1994, Volume 1, Issue 4 $2.50

Authoring Across the Rubicon: Font Issues

Authorware has a long history of supporting cross-platform development and delivery. Folks who remember *Course Of Action* may also remember *PC Transporter*. Despite a grand and glorious history of Mac to PC (now Windows) conversion and delivery, there are still some thorny issues for developers to resolve.

Since Macromedia has announced that the next version of Authorware will support bi-directional conversion (Mac to Windows / Windows to Mac), I thought this might be a good time to take a look at some of the common issues that need to be dealt with. Fortunately for us, Authorware handles the bit twiddling, palette re-mapping, format conversion, and other operations required of most media elements. (Did you know that black and white are at opposite ends of the Mac and Windows 256 color palettes? If you didn't know this, it's probably because Authorware does.)

Anyone who has done as many cross-platform development projects as we have can tell you that there is a long list of traps and pitfalls. For the sake of brevity (not to mention saving material for another AuthorWords issue), I will focus on fonts in this article. Future articles will address file names & paths, external files & formats, and user extensions (UCDs) implemented as XCMDs (Mac) or DLLs (Windows).

Long ago, we thought that Authorware had solved font problems for us by allowing packaging *with* fonts on the Macintosh. This spared us from having to install fonts in the user's System file/folder or resort to spooky and powerful utilities like ResEdit to install fonts directly into source files, packaged files or RunAPM. This worked great for the bit map fonts common at that time, but then came AP for Windows 1.0 and soon after that followed TrueType.

Our original problem was a configuration issue with bitmap fonts; did the end-user have all the fonts used by our application? Now we were faced with that dilemma again, but also had to deal with a file format conflict—Macintosh and Windows use different file formats for fonts. APW doesn't automatically convert Mac bitmap fonts into Windows bitmap fonts. Macromedia came to the rescue with a limited solution by providing pixel-for-pixel equivalents for the most common Mac fonts in common point sizes. But what about those of us with funky tastes, obscure Macintosh fonts, or special simulation requirements (e.g., a LED-like font)? We were left with two options, build these fonts ourselves (it is not so difficult as it is tedious), or bite the bullet and limit ourselves to the Mac-alikes provided with APW.

TrueType was heralded as a solution to the vexing problem of device independent font display and printing, but it only created more issues for multimedia developers like us. First, since TrueType fonts are algorithms rather than images, they are considered intellectual property and must be licensed accordingly. There were a lot of bitmap fonts that a developer could reasonably package with their AP applications without invoking the wrath of an attorney. However, most TrueType fonts really need to be licensed for re-distribution. As a result, if a corresponding bitmap font is available Authorware will attempt to embed it in lieu of the TrueType font. If a bitmap is not available, you will be warned, but Authorware will NOT embed the TrueType font resources. Furthermore, Microsoft and Apple render identical TrueType fonts slightly differently on the screen, so there may be some shifting (see figure 1). The most significant difference is in leading. Leading, the vertical space between lines of text, is named

(Continued on page 3)

A Brief Solicitation, Plea, and Request for Your Indulgence

Or: Why the $ in the Masthead?

We at AuthorWords originally intended this little rag to be a free and enjoyable service to our Authorware friends and family. Our readership, however, has outgrown our original intentions.

After much soul-searching, we have decided to introduce a shareware fee of $10.00 per year for AuthorWords. This will help us defray the cost of printing and mailing. We will not remove anyone from our mailing lists who doesn't pay (not right now, anyway). We are simply asking you to help us out if you find this newsletter valuable. Please make checks payable to Solis, and send them to 107 South B Street, Suite 350, San Mateo, CA 94401.

In a related matter, the rumor that we will accept bootlegged copies of *Dr. Quinn, Medicine Woman* in lieu of the ten dollars is absolutely not true.

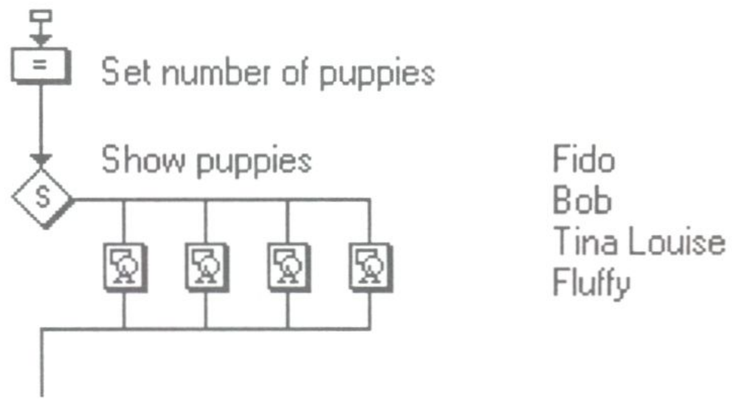
Beginner's Corner: Multi-Value Variables

If you are peering in this corner, you may not know what a variable is, so let's start with a working definition of a variable in Authorware. Simply put, a variable is a container. Just think of a variable as a box, like a mailbox, which has a name on the front. The name of the variable usually describes the kinds of values you might see inside the box. For instance, a variable named *Shoe Size* will contain values such as "4", "6", or "17EEE".

Authorware comes with a dazzling array of system variables. There are literally hundreds of built-in variables which are constantly being updated by Authorware; these variables allow authors to take a "snapshot" of the user's progress and activities at any time in an application.

For instance, to find out how many tries it took someone to correctly answer a multiple choice question, you could look at the value of the Authorware system variable *Tries*. Another example: the variable *WordClicked* will always contain the last word clicked on by the user, even if there is no click-touch area around the word. Take the time to browse through the Data - Show Variables menu to look at the full range of system variables.

An interesting and powerful variation on system variables is found in the following example. Let's assume that you have an interactive puppy adoption application, and you'd like to be able to get the litter count before your flow actually reaches the point where you display puppy photos:



The system variable PathCount might be useful here, because it contains the number of paths to the right of a decision icon. The following expression would be inside the calculation icon titled "Set the number of puppies".

Number in Litter := PathCount@"Show puppies"

The custom variable *Number in Litter* (you can create your own containers) is designed to contain the litter count. Notice that it is assigned by asking Authorware to look *ahead* down the flowline by specifying the decision branch titled "Show puppies". The system variable PathCount@"Show puppies" refers to the number of paths to the right of the *specified* decision icon.

Variables which get their values by specifying a particular icon by name are known as multi-value variables because they have different values depending on the icon to which they point. Approximately eighty-five variables in Authorware allow you to specify icons with the @"Icon Title" structure. Note that if you don't use the @"Icon Title" part of the variable, Authorware will assume you are pointing to the current, or most recent icon in the flow.

By the way, you can't *create* multi-value variables, although that would be a great feature to add to Authorware. For instance, you might create a custom variable called *Spayed*, and check for the value of *Spayed@"Fluffie"* (we hope that value is set to True!).

Ah well...to conclude this issue's Beginner's Corner, multi-value variables are powerful and useful tools, because they can give you specific information about what has been (looking back down the flowline), is now, and might be in the future (looking ahead down the flowline)! A little omniscience never hurt anybody...

Remember: You are not Alone

Don't forget that help for your Authorware travails is always available. Solis offers guided safaris down the flowline, as well as custom services. Our exclusive *Authorware Advanced Topics: Navigation and Tracking* is being offered soon in a city relatively near you. Call (415) 696-8700 for more information on expedition dates and times.

Herr Doktor Ikon Answers Your Authoring Questions for the Last Time

Doktor Ikon recently attended the Macromedia User Conference in San Francisco. He could be seen trudging through the halls, dragging his worn valise from session to session. What did he learn? A whole lot about performance tuning Authorware applications. We'll talk about that in a future version of AuthorWords. But as a hint of what is to come, the Doktor reveals that the Matte display mode is fastest on the Mac while the Opaque mode is quickest under Windows. That's great, but how do you implement two different modes in a cross-platform application? Good question. Very difficult. We'll have to think about it.

Now, the sad news. Doktor Ikon is leaving us. Extensive market research has shown us that his column is the least popular part of AuthorWords. The Doktor blames the lack of interest on short attention spans, an overworked readership, and talk radio.

Stay tuned to this page, however. The Doktor's replacement will be guaranteed to please. Look for cutting-edge commentary, industry gossip and rumors, Picks and Pans, and maybe even Scratch 'n' Sniff.

And to the few devoted fans of Doktor. Ikon, a bittersweet "Auf Wiedersehen."

Mmmm, Fonts

after the metal bars of lead once inserted between rows of type in the old fashioned print shop. Because of the differences in leading, bitmap overlays, or boxes that closely border text may not line-up properly on cross-platform applications.

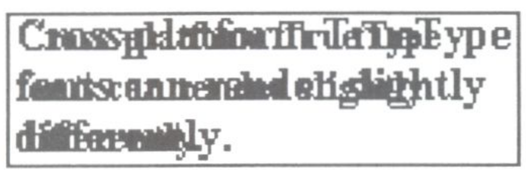
Let's summarize the issues: A.) Somehow we need to make sure users have the same fonts as developers, B.) Mac and Windows use different file format for bitmap fonts, C.) TrueType fonts can not be inserted into packaged Authorware applications, and D.) TrueType fonts render differently on Mac and Windows platforms. Given those four issues, let's look at some options you should consider. I will start with the more difficult to implement, but more accurate solutions. Later options will be easier to implement, but will sacrifice some fidelity.

Macintosh

Cross-platform TrueType fonts can render slightly differently.

Windows

Cross-platform TrueType fonts can render slightly differently.



Overlay of Mac & Windows
Times New Roman, Bold

Figure 1.

Option 1: Install bitmap fonts.

This option can be easily accomplished on the Macintosh by using only bitmap fonts and packaging *with* fonts. You could also use a utility like StuffIt InstallerMaker to install the required fonts in the user's System file or Fonts folder. For cross-platform projects using bitmap fonts means buying or making a pixel-for-pixel Windows bitmap font. You can create your own Windows bitmap fonts using tools from Microsoft's Multimedia Developer Kit and Software Developer Kit, MDK and SDK, respectively. A good Windows or Mac programmer should be able to create a small program that reads Mac NFNT/FONT resources, generates the intermediate *.fnt file, and links the object code stub necessary to create the final *.fon file. If any (all?) of that last sentence confused you, consider hiring a programmer or trying some of the options.

Advantages:

You can be assured that leading or character spacing problems will not make your text run off of the screen, nor will your nifty LCD font run right off the edge of your simulated digital voltmeter. In general, you own your font destiny. This can be fantastic if you have custom fonts or the skills to build them. It also allows you to accommodate issues like accent and bullet characters not being in the same location in Windows and Macintosh fonts. You may have seen this NOT working in cases where your lovely round Mac bullets ('•') show up as Yen ('¥') signs in Windows.

Disadvantages:

This option requires either a sophisticated developer / programmer to make fonts, or purchasing identical *bitmap* fonts for both platforms from a font vendor (often easier said than done). If you skip making a particular point size, you may end up with jagged edges on your text or misaligned text as the operating system tries to scale up or down from an available bitmap. Additionally, Windows fonts must be either installed manually on the end-user's machine, or installed via an installation/set-up utility, such as Eschalon Development's *EDI Install Pro*.

Option 2: Use TrueType fonts on both platforms, but get and use Mac bitmap fonts for final Mac packaging.

This option keeps all the advantages of Option 1, but sacrifices Windows pixel-for-pixel accuracy to eliminate the requirement of buying/building bitmap fonts on Windows. Apple, Microsoft, and others offer identical versions of TrueType fonts for both platforms. When developing your application, keep in mind the difference in Mac/Windows TrueType leading and allow additional vertical space beneath text.

Advantages:

You can be assured that leading or character spacing problems will not be a problem on the Mac. No worries about sharp jagged edges with your Windows fonts. Also, you do not have to create your own bitmap fonts or search for commercial fonts. If you do want to create your own fonts (or must do it), there are many powerful and easy-to-use tools for creating TrueType fonts. Most font development tools support PostScript and TrueType, but no longer support the proprietary bitmap format for output.

Disadvantages:

Windows fonts still must be installed either manually or via an installation utility (we are just not going to get away from that one). Also, you may still run

(Continued on page 4)

Fonts, Fonts, and More Fonts

(Continued from page 3)
into the typographical (not philosophical) 'bullets for Yen' syndrome. If this happens, just use APW's Find/Change function to replace all instances of '¥' with '•', and perform similar substitutions for other displaced characters/symbols. Finally, you may need to do additional 'touch-up' work on Windows to adjust for slightly taller/shorter or wider/narrower paragraphs of text.

Option 3: Use TrueType fonts exclusively on both platforms.

Now the Mac must bear the burden of an installer program, and you still have to deal with 'bullets for Yen' and leading problems.

Advantages:

No need to create bitmap fonts. Smaller Macintosh packaged applications. No jaggies on either platform. M flexibility for customization or ation of fonts.

Disadvantages:

'Bullets for Yen' syndrome, leading & wrapping problems. Installers required on both platforms. APW 'touch-up' probably required.

Option 4: Use APW's font mapping (stored in the APW.INI file) to map Mac fonts that come with the System to rough equivalents that are included with Windows.

For example, map Helvetica (Mac) to Arial (Win), Chicago (Mac) to System (Win), and so on. This will work with TrueType or bitmap fonts (or both mixed together). Unless your end-user has done something strange (like uninstall free fonts from Apple or Microsoft), you can be fairly sure they will see smooth type of an approximate size and shape you determined.

Advantages:

No need to buy or create *any* fonts. No installers required on either platform. Small(er) Macintosh packaged applications. Fewer jaggies on either platform.

Disadvantages:

'Bullets for Yen' syndrome, leading & wrapping problems. Limited selection of fonts. Fairly significant APW 'touch-up' work is likely to be required.

The options above are not the only alternatives, nor are they exclusive. For example, Macromedia provides 'Mac-like' fonts for Windows. A clever developer might limit their project to the subset of Mac fonts and point sizes that are included as Windows fonts with APW. This would allow you to use option 1 (and option 4), *without* requiring you to create any fonts. As always, the more simplifying assumptions you make, the easier it will be to successfully manage and implement your project. Keeping both this 'keep it simple' principle and good typographic design in mind, try to limit your application to 3-5 fonts. In most cases you will end up using something like 1-2 Sans Serif fonts for headings, call-outs, and control labels; 1-2 Serif fonts for body text, insets, and sidebars; and finally, 1-2 fonts to simulate digital displays, gauges, meters or whatever.

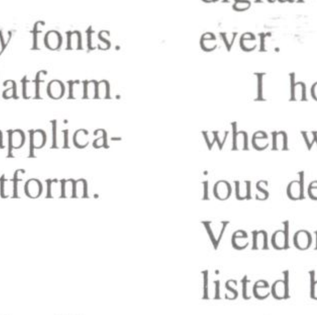
I hope you find this article useful when weighing the Pros and Cons of various development alternatives for fonts. Vendors for the products mentioned are listed below. Look for additional information on cross-platform issues relating to files, paths, and XCMDs/DLLs in upcoming issues of AuthorWords.

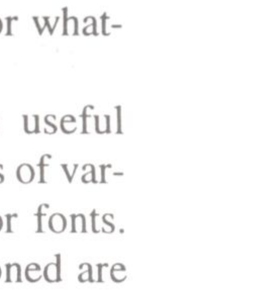
Products Mentioned in this Article
EDI Install Pro
Eschalon Development
(604) 945-3198

Installer Maker
Aladdin Systems
(408) 761-6200

AuthorWords

Solis
107 South B Street
Suite 350
San Mateo, CA 94401





Inside: The long lost Cross-Platform Issue, with more Authorware Professional™ tips, hints, and techniques.